**html5新特性**

**1.增型表单**

(1)新的input type <input type="?" />

(2)新的element

(3)新的属性--(\*)

1.1增型表单-新的**input type**

h4:button;submit;text;checkbox;radio;file

h5: search,tel,range,number;email;url;color;date;month;week

#建议:谨慎使用以上属性,兼容差

1.2:增型表单**-新元素**

h4:input;button;textarea;select;option;..

h5:datalist;progress;meter;output

-datalist 建议列表

当用户不清楚要输入内容时提供一些建议

<datalist id="d3"> 默认不可见

<option>xx</option>

<option>yy</option>

</datalist>

<input type="text" list="d3" />

-progress 进度条

显示一个进度条

<progress></progress> 左右晃动进度条

<progress value="0.7" /> 具有指定进度值的进度条

练习:使定时器+进度条实现一个可以

动态前进的进度条

创建定时器每隔1s修改进度

value=value+0.1

value==1 停止

-meter 刻度尺

<meter min="最小值" max="最大值" low="下限" high="上限" optimum="最佳值" value="当前值"></meter>

如果当前值低于下限刻度尺刻度尺为红色

如果当前值在下限刻度尺~上限刻度尺为黄色

如果当前值超过上限刻度尺为绿色

-output 输出(语义标签，单纯的显示文本)

没有任何外观样式,外观同 span

商品单价:￥3.5

商品数量:<input type="number" value="1" />

商品小计:<output>￥3.5</output>

1.3:增型表单-**新的属性**(重点\*\*\*\*\*)

<input ? />

h4:value;name;id;style;class;readyonly;

h5:

-autofocus 自动获取焦点

-placehodler 占位符

-form 用于把输入域放置在form外部，添加form属性的值与id相同，则也可以提交

<form id="f5">

</form>

<input type="number" name="age" form="f5" />

-与验证相关的属性：

required 不能为空

maxlength,minlength 最大字符串长度;最小字符串长度

min;max 数值最大值与最小值

pattern 正则表达式

**2.视频**

视频:创作生成视频,在网页中播放视频

Flash H5

Flash绘制(AS/FLex) Canvas+SVG

Flash绘画(游戏) Canvas+第三方游戏库

第三方游戏库: https://www.cocos.com/creator

Flash视频(音频) video/audio

Flash存储 webStorage

<video src="x.mp4"></video>

知识补充:

(1)常用视频格式:.mp4 .flv .webm

(2)如果浏览器播放指定视频，需要安装"解码器"

遇到问题不能播放

解决:使用工具软件"格式工厂" x.mp4->x.flv->x.webm依次选择

<video>

<source src="x.mp4" />

<source src="x.flv" />

<source src="x.webm" />

您的浏览器版本太低，请升级!

</video>

2.2:标签属性

-controls --是否显示控件 <video controls></video>

-autoplay --自动播放 <video autoplay></video>

#兼容性很差

-loop --是否循环播放 <video loop></video>

-muted --是否是静音播放

-poster --在播放之前显示广告图片 post="x.jpg"

-preload --预加载策略

auto:预先加载一定时长视频与元数据

metadata:预加载元数据(视频时长;尺寸;第一帧画面)——针对手机流量少的用户

none:不预加载数据

2.3:js属性

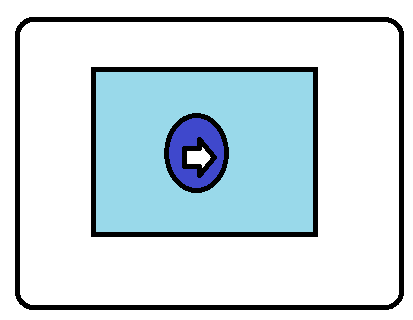
-volume:1 当前音量

-playbackRate = 5; 修改视频回放速率 大于1快放 小于1慢播

-paused 获取当前视频播放状态(true为暂停/false为播放)

-play() 播放视频(可在script中控制播放和暂停)

-pause() 暂停播放视频



2.4:**视频高级特性[事件]**--(重点)

-canplaythrough 当前视频加载结束可以播放会触发事件

属性：duration 获取当前视频总长度

-timeupdate 视频正在播放中(触发多次)

属性：currentTime 当前播时间点

-ended 播放结束

**如何隐藏video的下载按钮**  
1. 使用video元素的ControlList API  
<video controls controlsList="nodownload"></video>  
通过 ControList API，不仅可以通过设置nodownload来隐藏下载按钮，还可以设置nofullscreen隐藏全屏按钮, 以及noremoteplayback等属性，具体的示例可以参考github pages。该方案的优点是原生支持，不会产生bug，而且设置起来也是简单方便，不过由于 ControlList API 是从 Chrome 58+ 才开始引入的，所以对于低于 58 版本的 Chrome 只能通过方法2来隐藏了。  
2. 通过 CSS 隐藏  
通过Settings|Preferences -> Elements -> Show user agent shadow DOM打开 shadow DOM 开关后，可以通过 devtools 查看video元素的组成部分。  
由此发现下载按钮的是通过<input type="button" pseudo="-internal-media-controls-download-button"style="">呈现的，所以可以通过下述CSS将其隐藏：  
video::-internal-media-controls-download-button {  
 display: none;  
}  
问题：但是很可惜，该方法并不起作用，猜想可能是Chrome的一个BUG，

解决：简单来说就是将下载按钮挤出可见范围，这样就可以变相的认为按钮被隐藏了，代码如下：  
video::-webkit-media-controls {  
 overflow: hidden !important;  
}  
video::-webkit-media-controls-enclosure {  
 width: calc(100% + 32px);  
 margin-left: auto;  
}

video充满整个div标签播放：

css中——object-fit: fill; //全屏播放

**3.音频-(了解)**

3.1:html5新特性-音频

音频格式: x.mp3 x.wav x.ogg

<audio src="x.mp3"></audio>

#音频元素默认不可见，加controls

#属性和方法与视频大部分相同

练习:创建复选框控制背景音频播放

当复选框选中 背景音乐播放 play()

当复选框清空 背景音乐暂停 pause()

cb.change cb.checked==true

**4.绘图--canvas**

4.1:绘图

在网页中绘制图形:*股票走势图;销售统计图*;网页特效.网页游戏

网页中绘图技术三种

-SVG 2000年出现 2D矢量图

-Canvas 2014年出现 2D 位图

-WebGL 3D 位图,尚未纳入标准

#2D矢量图:使用线条绘制图形,可以无限放缩；

不失真,弱点颜色不够丰富

#2D 位图:使用像素点绘制图形,不可以放缩；

失真,优点颜色非常细腻

4.2:**矩形**

-通过标签创建画布

<canvas id="c3" width="500" height="400"></canvas>

注意事项:画布宽度和高度只能通过属性或者js来赋值；

不能使用css赋值.

-①通过js程序获取画布

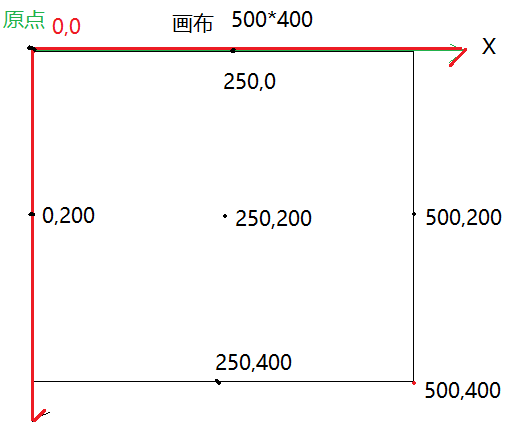
var c3 = document.getElementById("c3");

-②再依据画布获取画笔对象

var ctx = c3.getContext("**2d**");

#注意:一个画布只能对应一个画笔

-③绘图



①设置实心矩形样式(颜色)

ctx.fillStyle = "#f00";

-ctx.fillRect(x,y,w,h); 绘制实心矩形 x,y—左上角位置 w,h—宽度和高度

②设置空心矩形样式

ctx.strokeStyle = "#000";

-ctx.strokeRect(x,y,w,h); 绘制空心矩形 x,y—左上角 w,h—宽高

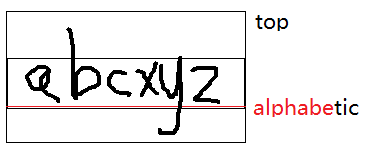
ctx.lineWidth = 1; 空心矩形边线宽度

ctx.clearRect(x,y,w,h); 清除矩形范围所有元素

#注意:坐标系

4.3:**文本**

-ctx.fillText(txt,x,y); x,y—左上角位置 txt—文本

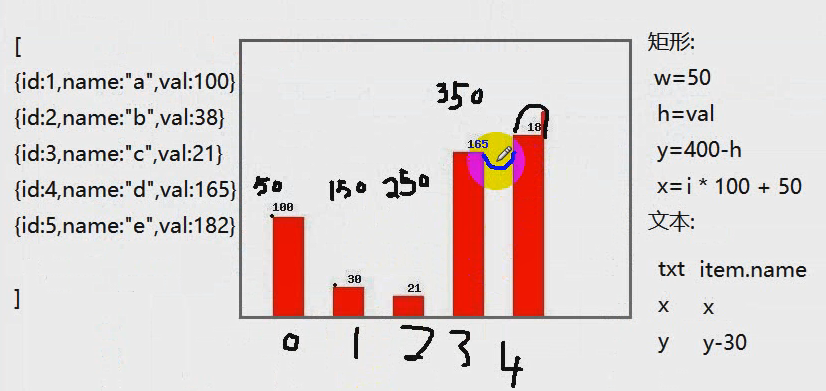
 ctx.font = "19px SimHei"; 字体大小与字体

ctx.textBaseline = "top"; 设置文本基线为top(默认基线在alphabetic在原点，看不见文本)

ctx.measureText(txt); 计算并获得文本宽度，返回js对象

如：{width:80}，通过属性 .width获得值

销售统计图：



4.4:**路径**

path:由多个坐标点组成任意图形，该图形默认不可见

可以使用 "描边"或"填充"实现

-ctx.beginPath(); 开始一条新路径

-ctx.closePath(); 闭合一条路径

ctx2.lineWidth="20"; //线宽

ctx2.lineCap="round"; //圆角

ctx2.globalAlpha=0.5; //透明度

-ctx.moveTo(x,y); 移动到一个指定点

-ctx.lineTo(x,y); 从当前点到指定点绘直线

-ctx.arc(cx,cy,r,start,end); 绘制圆拱形——从最右开始绘制

cx,cy 圆心

r 半径

start 起始弧度

end 终止弧度

#注意:绘制圆，使用弧度单位(不能使用角度单位)

角度: 0 180 360

弧度: 0 Math.PI 2\*Math.PI

#公式: 角度90\*Math.PI/180=弧度

-ctx.stroke(); 描边

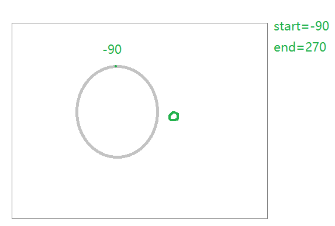
-ctx.fill(); 填充

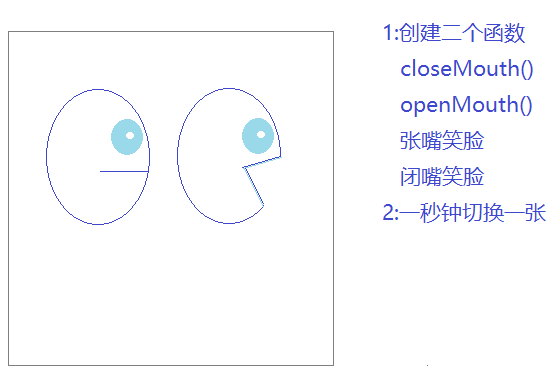
练习1:绘制可以动态前进的进度条(圆形)

起启角度:-90

终束角度:270

创建路径使用路径中圆拱形完成

 练习2:绘制二个动态笑脸



|  |
| --- |
| **智能定时器**(自动控制时间)：  function fn(){  requestAnimFrame(fn);  this.alpha+=0.0008\*36; //决定变化的快慢  }  fn();  var p=Math.sin(this.alpha) //取值-1~1  **贝塞尔曲线**(控制点到终点左右晃动)：  ctx2.quadraticCurveTo(x1,y1,x2,y2)  x1,y1 控制点坐标  x2,y2 终点坐标 |

4.5:**图片**

-canvas属于一种客户端技术[运行浏览器]

但是图片保存服务器

canvas绘制图片，需要先下载并且等下载结束，再绘制图片。

-图片为什么保存服务器

\*\*\*图片有版权

-操作流程

(1)创建图片对象

var p3 = new Image();

(2)下载图片对象

p3.src = "res/p3.png"

(3)绑定事件--(下载完成)

p3.onload = function(){

①绘制图片 ctx.drawImage(p3,x,y) 原始大小

②绘制图片 ctx.drawImage(p3,x,y,w,h) 拉伸(加了宽高—w/h)

p3.width p3.height //可以获得图片宽/高

}

4.6:**变形旋转**-(重点)

-变形旋转对应几个方法

ctx.rotate(弧度);

#默认旋转以画布原点为轴心(0,0)

#旋转角度会累加

-平移画布的原点

ctx.translate(x,y);

-保存画笔状态和恢复画笔状态

ctx.save();

ctx.restore();

|  |
| --- |
| <body>  <h1>2架飞机旋转</h1>  <canvas id="c2" width="500px" height="400px"></canvas>  <script>  var c2=document.getElementById("c2");  var ctx=c2.getContext("2d");  var p=new Image();  p.src="res/p4.png";  p.onload=function(){  // 创建两个变量，保存两架飞机旋转角度  var deg1=0;  var deg2=0;  var t=setInterval(function(){  // 保存画笔状态  ctx.save()  // 平移原点  ctx.translate(100,100);  // 旋转角度  ctx.rotate(deg1\*Math.PI/180);  // 绘制第一架飞机  ctx.drawImage(p,0-p.width/2,0-p.height/2);  // 恢复画笔状态  ctx.restore();  // 旋转角度  deg1+=15;  // 第二架飞机  ctx.save()  ctx.translate(400,100);  ctx.rotate(deg2\*Math.PI/180);  ctx.drawImage(p,0-p.width/2,0-p.height/2);  ctx.restore();  deg2+=30;  },100)  }  </script>  </body> |

4.7:**渐变对象**

渐变对象:是一种图像效果，从一种颜色慢慢过渡到另一种颜色.

-创建渐变对象

var g = ctx.createLinerGradient(x1,y1,x2,y2);

x1,y1 起点坐标

x2,y2 终点坐标

-添加颜色点

g.addColorStop(offset,color);

offset 偏移量 0 0.5 1

color 颜色

-将渐变对象赋值描边样式或者样式

ctx.fillStyle = g;(填充)

ctx.strokeStyle = g;(描边)

-绘制图

ctx.fillRect(x,y,w,h);

**5.绘图-svg**

5.1:canvas--svg

|  |  |  |
| --- | --- | --- |
|  | canvas | svg |
| 绘制图像类型 | 2D位图 | 2D矢量 |
| 如何绘图 | 使用js代码 | 使用标签绘图 |
| 事件绑定 | 只能将事件绑定画布 | 每个图形都可以绑定事件 |
| 应用场景 | 网页特效,小游戏 | 地图 |

5.2:svg -开发流程

-1:创建画布

<svg id="s3" width="500" height="400"></svg>

-2:在画布中添加标签

<svg id="s3" width="500" height="400">

<rect x="" y="" width="" height="" fill="" fill-opacity="" stroke="" stroke-opacity=""></rect>

</svg>

fill：填充颜色

|  |
| --- |
| svg绘图特性  (1)svg图形样式可以使用 css声明  但是样式只能写svg属性不能用css属性  示例:  #s3{  border:1px solid red; //不对  stroke:1px; //正确  }  (2) svg动态创建图形  -字符串拼接  var html = "<rect></rect>";  svg.innerHTML = html;  -创建元素对象  var rect = document.createElementNS("http://www.w3.org/2000/svg",**"**图像标签名称**"**);  rect.setAttribute(“属性名”,值);  svg.appendChild(rect);  (3)图形使用js 赋值  #不能使用html dom方法,只能使用核心DOM方式  rect.x = 100; rect.y = 80; 错误  rect.setAttribute("x",100);  rect.setAttribute("y",80); |

5.3：**圆形**

<circle cx="" cy="" r=""></circle>

创画布上创建 30个大小随机，位置随机，颜色随机,

透明度随机圆形

当用户点击某个圆形，当前被点击圆形慢慢变大变淡

直至消失,从DOM树删除

5.4：圆形-**椭圆,直线**

<ellipse rx="" ry="" cx="" cy=""></ellipse>

rx:ry 水平半径 垂直半径

cx;cy 圆心

<line x1="" y1="" x2="" y2=""></line>

x1,y1 起点坐标

x2,y2 终点坐标

stroke-width 边线宽度

stroke-linecap 顶端外观 round

stroke 边线样式

5.5:**折线,文本**

一条折线上可以有任意多个连续点(x,y)

<polyline points="x,y x,y ..." stroke="red" fill="transparent"></polyline>

<text font-size="" fill="" stroke="" x="" y="">文本的内容</text>

5.6:**图形**

<image xlink:href="图片路径" x="" y="" width="" height="></image>

5.7: **渐变对象**

<defs> 特效对象

<linearGradient id="g3" x1="" y1="" x2="" y2="">

<stop offset="" stop-color="">

</linearGradient>

</defs>

x1,y1 起点坐标 0% 100%

x2,y2 终点坐标 0% 100%

offset 偏移量 0% 50% 100%

stop-color 颜色点

<ANY fill="url(#g3)" stroke="url(#g3)"></ANY>

5.8:**滤镜**

<defs>

<filter id="f3">

<feGaussianBlur stdDeviation="3" />

</filter>

</defs>

# stdDeviation="3" 模糊度(越大越模糊)

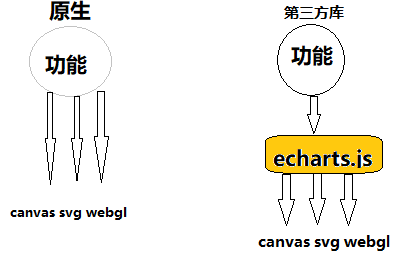
<ANY filter="url(#f3)"></ANY>

所有滤镜列表

https://developer.mozilla.org/zh-CN/docs/Web/SVG/Element/filter

**6.echarts--**百度提供第三方网页绘图库(对象;函数)

(1)工作方式



(2)下载

https://echarts.baidu.com/download.html

(3)了解echarts案例

(4)常规使用 ----echarts--入门示例-**柱状图**

-创建存储图形容器

<div id="main" style="width:500px;height:400px;"></div>

-将echarts.js 文件引入当前html文件

-通过js获取容器并且创建echarts对象

var main = document.getElementById("main");

var mychart = echarts.init(main);

-通过js创建配置对象并且将配置对象添加echarts对象中

var option = {

title:{text:"入门示例"}, 标题

xAxis:{data:["北京","上海","广州","深圳"]}, x轴数据

yAxis:{}, y轴数据

series:[{type:"bar",data:[100,110,50,80]}] 数据源

};

mychart.setOption(option);

(5)常规使用 ----echarts—**线状图**

var option = {

title:{title:"折线图"},

xAxis:{data:[".."]}.

yAxis:{},

series:[{type:"line",data:[]}]

};

(6)常规使用 ----echarts—**饼状统计图**

var option = {

title:{text:"饼状统计图"}

series:[{

type:"pie", //饼状图

center:["50%","50%"] //圆心

radius:"50%" //半径

data:[

{value:335,name:"北京"},

{value:315,name:"上海"},

{value:135,name:"广州"},

]

}]

}

(7)常规使用 ----echarts--gauge

**7.WebWorker-(代码3行)**

程序:指可以被CPU执行代码,存储在磁盘。

1.html 1.css 1.js...

进程:指程序被OS调用内存中并且分配执行空间。

线程:线程是进程内部组成部分。

chrome浏览器线程模型：

一个chrome进程内部至少有6个线程，向web服务器发送请求获取资源--资源请求线程；

还有一个线程负责所有资源绘制并且负责执行；

js程序--UI主线程。

示例：

<button>按钮一</button>

<script src="1.js"></script>

<button>按钮二</button>

问题：js耗时较长，btn按钮会延迟显示

原因:出现以问题:执行js和绘制按钮同一个UI主线程

解决方案:创建一个新线程由它执行耗时js任务

<button>按钮一</button>

<script>

new Worker("1.js");

</script>

<button>按钮二</button>

#(1)创建Worker对象

#(2)创建新线程执行1.js

#注意事项:浏览器不允许Worker线程执行程序中包含任何DOM/BOM对象，也就是不能执行类似JQuery程序。

|  |
| --- |
| 作用:如果有一个程序出现大量耗时计算同时没有DOM/BOM代码，可以使用Worker执行此程序。  通常:复杂计算, 大量统计可以使用Worker。 |

7.2:html5新特性--WebWorker-代码3行

-UI发送数据给Worker

#ui 发(05.html中)

var w = new Worker("05.js");

w.postMessage(stringMsg)

#Worker 收(05.js中—接收并处理数据）

onmessage = function(e){

e.data

}

-Worker发送数据UI

#worker发数据(05.js中)

postMessage(stringMeg)

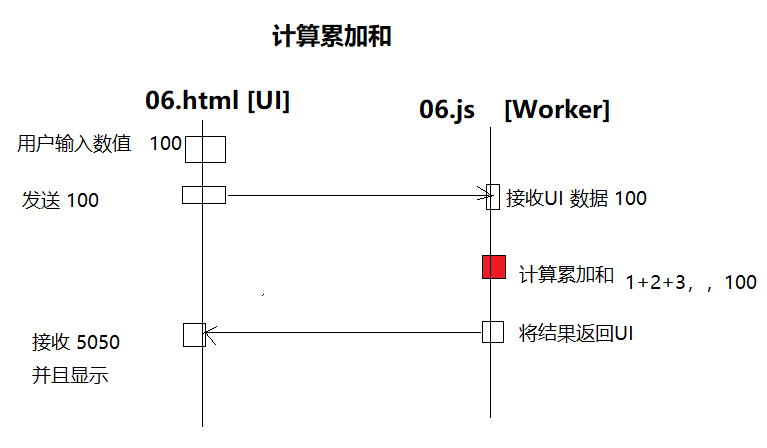
#ui 接收(05.html中)

w.onmessage = function(e){

e.data

}

小结:webWorker是一个新线程，用于执行耗时js任务

练习:

\*1:创建网页 06.html 添加input "请输入需要计算数据"

\*2:在 input下添加按钮"开始计算累加和"

\*3:为按钮绑定点击事件 ,创建Worker执行 06.js

4:Worker计算累加和并且并结果发送UI 06.html

6:06.html 负责显示结果

|  |
| --- |
| 04.html  <script>  var num=document.getElementById("num")  var btn=document.getElementById("btn")  var d4=document.getElementById("d4")  btn.onclick=function(){  var w=new Worker("04.js")  w.postMessage(num.value)  w.onmessage=function(e){  d4.innerHTML=e.data;  }  }  </script> |
| 04.js  onmessage=function(e){  for(var i=1,sum=0;i<=e.data;i++){  sum+=i;  }  postMessage(sum)  } |

**8.WebStorage--本地储存**

8.1:WebStorage功能在浏览器中存储当前用户专用数据,

购物车,定制样式..

#哪些技术可以让数据存储在客户端：

-cookie:兼容性好,不能超过4kb,操作复杂；

-Flash:依赖Flash播放器；

-WebStorage:不能超过8MB，操作简单；

-IndexDB:可以存储大量数据，还不是标准技术。

**8.2:sessionStorage**

将数据保存sessionStorage,在当前会话中所有网页，都可以读取或修改数据,但是一但会话结束，sessionStorage保存数据丢失。

#建议保存数据:用户昵称;定制样式;

#建议:客户端保存数据一定是安全要求低

-保存数据

sessionStorage["key"]="value"; #key数据名 value数据值

sessionStorage.setItem("key","value");

示例:

sessionStorage["uname"]="狗蛋";

#注意事项:key不能重复,如果重复被覆盖！

sessionStorage["uname"]="二蛋"

-获取数据

数据值：

var value = sessionStorage["key"];

var value = sessionStorage.getItem("key");

数据名：

var key = sessionStorage.key(i); i为session中存储的下标，0开头

示例:

var "狗蛋"= sessionStorage["uname"];

-清空数据

sessionStorage.removeItem("uname"); 删除一组数据

sessionStorage.clear(); 删除所有数据

-其它数据

sessionStorage.length; 几对数据

练习:用户登录判断

#此练习只是为了熟悉sessionStorage使用，项目中用户登录

一定服务器session对象判断用户是否登录

-创建 03\_index.html 首页

04\_login.html 登录网页

05\_logout.html 退出网页

-03\_index.html 右上角提示"请登录" 超链接

-04\_login.html 创建表单，用户输入用户名密码登录按钮

用户点击按钮 '登录成功 3s 跳转首页'

sessionStorage.setItem("uname","xxx")

setTimeout + 3 + location.href

-03\_index.html "欢迎回来 xxx 退出"

-05\_logout.html 退出成功 3s 返回首页

-03\_index.html "请登录"

首页-->登录—>首页-->退出-->首页

|  |
| --- |
| 02\_index.html  <body>  <h1>首页</h1>  <div id="d1">  <a href="02\_login.html" id="a1">请登录</a>  </div>  <script>  var uname=sessionStorage.getItem("loginName");  //判断是否登录保存数据(用户名)  if(uname){  var html=`欢迎${uname}回来`;  html+="<a href='02\_logout.html'>退出</a>"  var d1=document.getElementById("d1")  d1.innerHTML=html  }  </script>  </body> |
| 02\_login.html  <body>  <h1>登录页</h1>  <form action="">  用户姓名：<input type="text" name="uname" id="uname"><br>  用户密码：<input type="password" name="pwd"><br>  <input type="button" value="登录" id="btn">  </form>  <script>  var unname=document.getElementById("uname")  var btn=document.getElementById("btn")  btn.onclick=function(){  var n=uname.value;  sessionStorage.setItem("loginName",n)  alert("3s后跳转回首页")  setTimeout(function(){  location.href="02\_index.html"  },3000)  }  </script>  </body> |
| 02\_logout.html  <body>  <h1>退出页</h1>  <script>  sessionStorage.removeItem("loginName");  alert("3s后跳转到首页");  setTimeout(function(){  location.href="02\_index.html"  },3000)  </script>  </body> |

**8.3:localStorage**

将数据保存localStorage在当前会话中和后继所有会话中所有网页都可以读取或修改数据

#localStorage 相当于永久保存

#建议保存数据:用户IP/购物车 ;

#建议:客户端保存数据一定是安全要求低

保存(修改，相同名字key会覆盖)

localStorage[key] = value;

localStorage.setItem(key,value)

获取

var value = localStorage[key];

var value = localStorage.getItem(key);

var key = localStorage.key(i);

删除

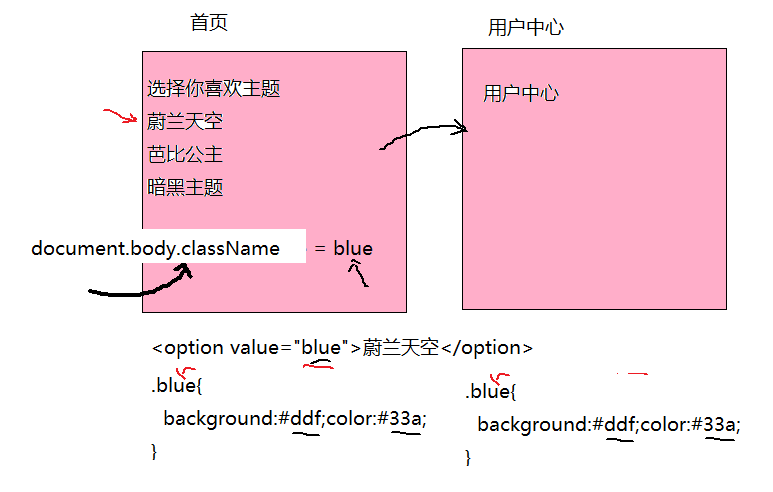
localStorage.clear();

localStorage.removeItem(key)

其它

localStorage.length;

练习：用户自定义的主题



-创建06\_index.html 首页并且在首页创建下拉菜单

"请选择你喜欢的主题"

"蔚兰天空" "芭比公主" "暗黑主题"

-网页head添加样式

.blue{background:#ddf;color:#33a}

.pink{background:#fdf;color:#a3a}

.dark{background:#333;color:#eee}

每个下拉菜单 value 与 class 名称相对

当用户选中某个项,获取当前value

为当前body元素指定样式

var c = this.value;

document.body.className = c;

#注意事项:

localStorage如果修改数据setItem()——value发生变化

修改保存可以立即触发事件:

window.onstorage=function(){ ... }

sessionStorage 保存数据如修改不会触发onstorage事件

此事件目的:监听此事件发现 localStorage数据是否被改变了

* 小结:WebStorage 提供二个对象 sessionStorage/localStorage

都是在客户端存储数据,

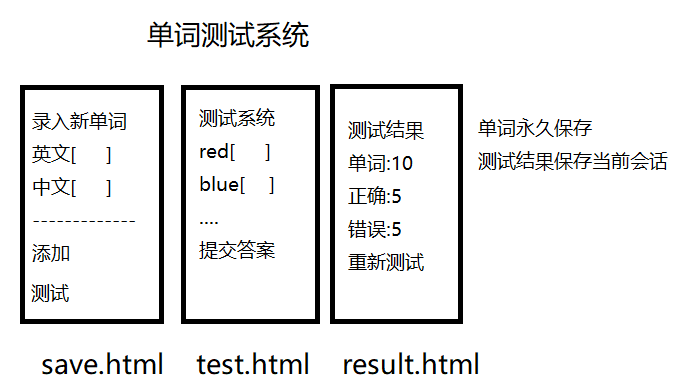
sessionStorage 将数据保存在当前会话有效

localStorage 将数据保存在永久范围

练习:单词测试系统

(1)创建save.html 网页录入新单词

(2)创建test.html 开始测试

(3)创建save.html 显示测试结果

提示:单词永久保存 localStorage

测试结果 sessionStorage

|  |
| --- |
| 保存单词：  <body>  <h1>单词录入：永久保存</h1>  <form action="#">  英文：<input type="text" id="eg"><br>  中文：<input type="text" id="ch"><br>  <input type="button" value="保存" id="btn">  <a href="05\_test.html">开始测试</a>  </form>  <script>  var eg=document.getElementById("eg");  var ch=document.getElementById("ch");  var btn=document.getElementById("btn")  btn.onclick=function(){  var e=eg.value;  var c=ch.value;  console.log(e+"\_"+c)  localStorage.setItem(e,c);  eg.value="";  ch.value="";  }  </script>  </body> |
| 测试系统：  <body>  <ul>  <!-- <li>  <span>red</span>  <input type="text" id="red">  </li>  <li>  <span>blue</span>  <input type="text" id="blue">  </li> -->  </ul>  <button id="btn">提交试卷</button>  <script>  sessionStorage.clear();    var ul=document.getElementsByTagName("ul")[0];  var html="";  for(var i=0;i<localStorage.length;i++){  var key=localStorage.key(i);  console.log(key);  console.log(localStorage[key]);  html+=`<li><span>${key}</sapn><li><input>`;  ul.innerHTML=html;  }  var btn=document.getElementById("btn");  btn.onclick=function(){  // sessionStorage.setItem(key);  var pt=document.getElementsByTagName("input");  console.log(pt)  console.log(pt.length)  var num=0;  for(var i=0;i<pt.length;i++){  var key=localStorage.key(i);  console.log(pt[i].value)  if(pt[i].value==localStorage[key]){  num++;  }  }  var score=eval(num/pt.length\*100).toFixed(2);  sessionStorage.setItem("分数",score)  location.href="05\_result.html" //跳转到05\_result.html  }  </script>  </body> |
| 获取分数：  <body>  <div id="d1"></div>  <script>  var d1=document.getElementById("d1");  var score=sessionStorage.getItem("分数");  if(score==null){  d1.innerHTML="请提交测试"  }else{  d1.innerHTML=`您的分数是${score}分`  }  </script>  </body> |

**9.WebSocket--实时通讯**

HTTP协议:

HTTP协议作用:用于传输网页资源(html;css;js;img;mp4;mp3..)

HTTP 搬运工

HTTP工作模型:"请求-响应",客户端发送给服务器数据请求,

服务器发送客户端数据响应

一次请求只对应一次响应,没有请求就没响应

此协议在有一些应用场景下就力不从心

任务:股票走势图

解决方案:启动新协议: WebSocket

WebSocket:工作模型“广播和收听”客户端连续服务器上就不再断开,永久连接双方都可以随时发送消息。

9.1:代码

-ws服务器 node.js/java/php

#监听指定端口 9001

#接收客户请求

#向对方发送消息

#或者接收

-ws 客户端对象

#创建ws对象

var ws = new WebSocket("**ws**://127.0.0.1:9001");

#向服务器发送数据

ws.send(stringMsg)

#接收服务器数据

ws.onmessage = function(e){e.data}

#断开连续

ws.close();

小结:

(1)哪些应用场景需要WebSocket 服务器发送数据，不定量

比如股票之类

(2)聊天室

**10.拖放API**

Drag & Drop:拖动和释放

-拖动的源对象(图片-会动)-触发3个事件

dragstart 拖动开始

drag 拖动中

dragend 拖动结束

整个过程:dragstart\*1+drag\*n+dragend\*1

-拖动的目标对象(垃圾桶-不会动)

dragenter 拖动进入

dragover 拖动悬停

dragleave 拖动离开

drop 在上方释放

整个过程1:dragenter\*1+dragover\*n+dragleave\*1

整个过程2:dragenter\*1+dragover\*n+drop\*1

注意:必须阻止dragover的默认行为drop才可能触发

练习:拖动删除效果



|  |
| --- |
| 拖动删除  <body>  <img src="res/t.png" alt="" id="t">  <hr>  <div class="container">  <img src="res/p3.png" alt="" id="p3">  <img src="res/p4.png" alt="" id="p4">  <img src="res/p5.png" alt="" id="p5">  </div>  <script>  // 创建全局变量pid  var pid="";  // 获取小飞机们  var imgs=document.getElementsByClassName("container")[0].children;  // 创建循环，为小飞机绑定拖动事件]  for(var img of imgs){  img.ondragstart=function(){  // 拖动开始时，记录当前小飞机id  pid=this.id;  console.log(pid)  }  }  // 获取回收站  var t=document.getElementById("t");  // 为回收站绑定4个事件  t.ondragenter=function(){}  t.ondragover=function(e){  e.preventDefault();  }  t.ondragleave=function(){}  // 其中在释放中依据pid删除指定小飞机  t.ondrop=function(){  var m=document.getElementById(pid)  m.remove();  }  </script>  </body> |

**11.地理定位(概念-使用百度地图)**

11.1:Geolocation:地理定位,使用js获取当前浏览器所在地理(经度/纬度/海拨/速度)数据,用于实现LBS(location Based Service)基于位置服务.

示例:饿了么,滴滴打车,高德导航...

手机如何获取定位信息：

(1)首先手机GPS芯片网络通信,定位精度米范围；

(2)次选手机通信基站定位获取，精度公里范围。

HTML5中提供一个对象,用于获取当前浏览器信息

window.navigator.geolocation{...}

11.2:第三方定位服务

#百度地图

#高德导航

#腾讯地图

11.3:地图开发流程

-打开百度地图开发者平台,注册帐户

http://lbsyun.baidu.com/

-为你的网站请求百度访问密钥：AccessKey

-在网站中嵌入百度地图代码

(1)创建地图实例

var map = new BMap.Map("container");

(2)创建坐标点

var point = new BMap.Point(经度,纬度);

(3)以坐标点为中心显示地图

map.centerAndZoom(point,15);

15—显示级别，取值：1-19

**12.大鱼小鱼-(canvas)**

#分析游戏功能:

大鱼和小鱼,大鱼吃食物,吃蓝色食物100分，吃到橙色食物200

海葵50条(难点海葵摆动)

食物30食物其中只显示15个(小难点食出生)

#分析游戏目录结构:

game 游戏所有内容保存在此目录

js 目录

commonFuntions.js 游戏函数文件

#src 目录图片(67图片)

大鱼眼睛 bigEye0.png bigEye1.png

大鱼身体 bigSwim0.png ... bigSwim7.png

大鱼尾巴 bigTail0.png bigTail7.png

背景 background.jpg

食物 fruit.png blue.png

小鱼眼睛 babyEye0.png babyEye1.png

小鱼身体 babyFade0.png ... babyFade19.png

小鱼尾巴 babyTail0.png .. babyTail0.png

#index.html 唯一网页

功能一:index.html

<canvas></canvas> 前 800\*600 大鱼;小鱼;分数...

<canvas></canvas> 后800\*600 背景;海葵;食物

功能二:加载js文件 commonFunction.js

游戏函数库文件

<script src="js/commonFunction.js"></script>

功能三:创建并加载js/main.js 程序入文件

作用:创建全局变量

创建游戏中角色对象

绘制角色对象

功能四:创建背景角色

(1)图片src/background.jpg

(2)main.js 全局变量 bgPic/init 创建图片对象下载图片

(3)创建background.js

function drawBackground(){}

(4)在gameloop创建定时器

调用drawBackground()

功能五:定时器

function fn(){

setTimeout(fn,100)

console.log(123);

}

fn();

智能定时器使用方法

function fn(){

requestAnimationFrame(fn);

ane.drew();

}

fn();

requestAnimationFrame兼容性差

帮忙出现 commonFunction.js

功能六:海葵ane.js 不动

功能七:海葵ane2.js 会动

|  |
| --- |
| ane.js中  // 2.4 为aneObj添加绘制方法  aneObj.prototype.draw=function(){  // 2.4.1 保存画布状态  ctx2.save();  ctx2.strokeStyle="#3b154e"; //紫色  ctx2.lineWidth="15"; //线宽  ctx2.lineCap="round"; //圆角  ctx2.globalAlpha=0.5; //透明度  // 2.4.2 累加非常小数值，返回-1~1数值  this.alpha+=0.0008\*36;  var p=Math.sin(this.alpha)  // 2.4.3 创建循环，画50个海葵  for(var i=0;i<this.num;i++){  // 2.4.4 开始新路径  ctx2.beginPath();  // 2.4.5 计算终点坐标  this.headx[i]=this.rootx[i]+this.amp[i]\*p;  // 2.4.6 移动起点坐标  ctx2.moveTo(this.rootx[i],canHeight)  // 2.4.7 绘制贝塞尔曲线，控制点，终点  ctx2.quadraticCurveTo(this.rootx[i],canHeight-100,this.headx[i],this.heady[i])  // 2.4.8 描边  ctx2.stroke();  }  // 2.4.9 恢复画笔状态  ctx2.restore();  } |